International Journal of Engineering & Technology, 7 (3.4) (2018) 52-56

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper



Depth Impurity Pruned Strategies for Extracting High Utility Itemsets

K. Santhi1*, B.Valarmathi[2], T.Chellatamilan[3]

[1] *Department of Analytics,School of Computing Science and Engineering (SCOPE), VIT University, Vellore,India*
[2]*Department of Software Systems,School of Information Technology and Engineering(SITE), VIT University, Vellore,India*
[3]*Department of Computer Science and Engineering,Arunai Engineering College,Thiruvannamalai, India*
**Corresponding author E-mail: santhikrishnan@vit.ac.in*

Abstract

Normally in a transaction database mining high utility itemsets indicates to the location of itemsets which is causing high utility like benefits. In spite of the fact that various important calculations have been proposed as of late, they bring about the issue of generating a huge amount of itemsets for mining to discover HUI. Mining is reduced by such an extended quantity as far as execution time and space complexity. When the database contains large amount of transactions, this condition may turn into mediocre. In this research paper, we account this concern by offering a state-of-the-art calculation named Depth Impurity Quality Index Pruned strategies which considers the complexity of sub-trees to more efficiently identify high-utility itemsets. It is an collection of common itemset which are used for mining and is significantly harder, inflexible. This is imputable to the absence of intrinsic organizational behaviour of HUI which could have worked. This paper suggests a high utility mining technique which make use of novel pruning approaches.The experimental outcomes disclose that the proposed method is exceptionally viable in killing unhopeful applicants in the database transactions.

Keywords:*Depth Impurity;Execution time; high utility itemset mining;Itemset; Pruned strategies*

1. Introduction

HUI mining is typically because of its use in numerous lines and research purpose applications. For a decision maker mining of HUI make available freedom to include the notion of item utilities into the itemset mining process. To support effective mining, mining methods utilize property known as anti-monotonic. High Utiity mining becomes a huge problem because of this. Transaction weighted utility idea is used by the algorithms[8], which is used to reduce the number of computations in a mining process. But transaction weighted utility overcalculate itemset's utility. This causes huge wastage of computations for the itemsets, which will not satisfy the minimum utility threshold in the end. In this paper we try to discuss about the limitations using a new model for High Utility Mining which incorporates many strategies.

We illustrate the benefits of the newly suggested model with the help of performing various experiments on numerous datasets.

The remaining sections are organized in the following manner. Second section discusses the associated work of HUI mining process. Section 3 describes the key definitions with notations used.

Section 4 discribes the proposed depth impurity quality index-pruning strategies. Section 5 provides a experimental results and analysis. Finally, Section 6 provides conclusion and remarks.

2. Literature Survey

Lin, J. C. W., Ren, S.[1] and Liu et al. [8] use the idea of TWU, which is used for HUI mining from the database by computing authentic itemsets utilities are calculated and itemsets with low utility values are rejected and then Yao and Hamilton [3] discussed two pruning structures by having utility and support upper limit. Degree-wise mining is required because pruning properties are being neglected by the algorithm. The mining method is used to generate a constricted utility upper bound by removing unnecessary items sequentually before conducting computation utility to set a more rigorous upper limit. Li et al., [4] proposed algorithms which use an remote item removal strategy to hold the quantity of candidates brought forth and tested in level-wise utility mining.

Ahmed[5] proposed three noval tree structures effective high utility. Lin, J. C.Vincent[6] presented 2 algorithms known as UP Growth and UP growth$^+$ for efficient extraction of HUI. A comparision between the UP growth and state of art algorithms is presented. Liu, M., &Qu, J[7] presented an algorithm for pruning the search space of HUI-Miner.Lan, Hong, & Tseng[9] proposes an algorithm which is an extension of the two-phase algorithm (Liu et al., 2005)[8].

Lan, Hong, & Tseng, 2014 [10] follows PB algorithm approache to proficiently mine HUI. PB algorithm requires less computations .(Erwin, 2007)[11]. Ronghui Wu1 and Zhan He1, 2018 [12] developed a framework of top-k high average-utility itemsets mining instead of setting a minimum average-utility threshold.

Liu, Wang, and Fung (2012)[13] proposes d2 HUP algorithm which uses ideas such as unnecessary, pruning for performing mining of HUI efficiently. The enhanced versions of HUI-Miner (Krishnamoorthy et al, 2015)[14] FHM (Fournier-Viger et al, 2014)[16] and EHAUPM: (Lin, J. C. W., Ren, S., Fournier-Viger et al, 2017)[1] were proposed. Even after all the research efforts, HUI requires costlier computaions in memory usage and execution

Copyright © 2018 Authors. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

time.[14][15].(Fournier-Viger et al, 2014; Krishnamoorthy et al, 2015).This exploration work improves the existing methodologies and offers fresh plans for mining HUI efficiently.

3. Definition and Notation

Definition 3.1: Let $I=\{i_1,i_2 \ldots i_m\}$, this set contains distinct elements. A itemset is a set X which is subset of I ($X \subseteq I$). A transaction T_j may have multiple items $T_j=\{x_l | l=1,2 \cdots N_j, lx \in I\}$, N is items. Database $D=\{T_1,T_2 \ldots T_n\}$, n is transactions present inside database. Table 1 shows how the transactions work .

Definition 3.2: Each item has a utility value, shown in Table 2. Also, each item has an internal utility value.

Table 1. A transaction database

TID	Transaction	Purchase Unit (IU)	Utility value (U)	Transaction utility value (TU)
T1	{d, c, a}	{1, 1, 1}	{2,1,5}	8
T2	{g,e,c,a}	{5,2,6,2}	{5,6,6,10}	27
T3	{f,e,d,c,b,a}	{5,1,6,1,2,1	{5,3,12,1,4,55}	30
T4	{e,d,c,b}	{1,3,3,4}	{3,6,3,8}	20
T5	{g,e,c,b}	{2,1,2,2}	{2,3,2,4}	11

The following are utility values of the items

{g,f,e,d,c,b,a}={1,1,3,2,1,2,5}

Definition 3.4. The item utility X_i represented as $U(X_i,T_j)$ and is given by

$$U(X_i,T_j)=EU(X_i) \text{ x } IU(X_i,T_j)$$

.Example 2.2. $U(b, T_4) = 2 * 4 = 8$.

Definition 3.5. The itemset utility X of a T_j is $U(X,T_j)$ and is given by $U(X,T_j) = \sum U(i,T_j)$ if X is subset of T_j. Else $u(X,T_j) = 0$.

$$U(\{b,c\},T_5) = U(b,T_5)+U(c,T_5) = 2*2+1*2=6$$

Definition 3.6. U(X) is the itemset utility X and defined as

$$U(X) = \sum U(X_C,T_C)$$

Every Transaction T_c must be in g(X) where g(X) consists of all the transactions of X.

Example 2.3 itemset {a,c} utility is given by

$$U\{a,c\}=U(\{a,c\},T_1)+ U(\{a,c\},T_2)+ U(\{a,c\},T_3)$$

$$= U(\{a,T_1\})+ U(\{c,T_1\})+ U(\{a,T_2\})+ U(\{c,T_2\})+ U(\{a,T_3\})+ U(\{c,T_3\})$$
$$= 5+1+10+6+5+1=28$$

Definition 3.7. An itemset X is called HUI if its utility u(X) is greater than the threshold mentioned by the user Else, X is a low-utility itemset.

HUI = { X: U(X) / X subset of I, U(X) $\geq$ min util}

Example 3.8. The high-utility itemsets in the database of the example when minutil = 32 are shown in Table 3.

Table 3. High-Utility itemsets

Itemset	Utility
{d,c,b}	34
{e,d,b}	36
{e,d,c,b}	40

4. Data Structure and Key Pruning Strategies

In DIQIP, branch and bound technique are implemented for finding the optimal solutions in operation research. Getting down from the fine predicted answer, in the root of the branch and bound tree, towards the best exact solution. In the case of maximization problem which consists of three primary parts.

1. A minimum utility value as an upper bound for the best solution in the search space is provided by a "bounding function''.
2. The next promising nodes, which are to be explored is chosen by a "selection function''
3. A ''branching rule'' that splits by generating the children of a node.

An ''initial solution'' can be used in the branch and bound tree in order to avoid unnecessary branching of nodes If is near to the better solution it help to lessening the number of nodes. If during the search process an improved solution than the current best solution is established then the current best solution is reorganized. Termination in search has dual ways. First one, it terminates when all promising nodes have been updated. Second one, it breaks when the user quantified depth limit has been surpassed. The amount of branching at a particular search level is controlled indepth impurity pruning approach. In the issue of tree depth, our heuristics allow us to quickly find local feasible solutions and once the feasible solution has been obtained, then at the next depth level some nodes are explored.

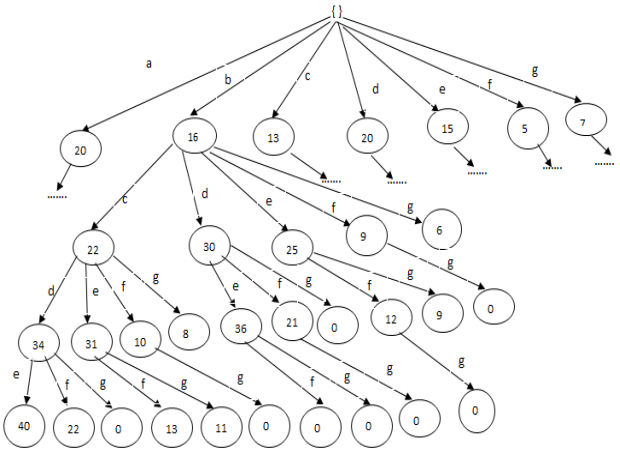


Figure.1 A model tree with design when no pruning is applied

In the branch and bound technique a node τ is represented by τ (ϕ, I, s), in which ϕ is to represent the value its stores, s represents the minimum utility value, and I represent the index value of the transaction .

Definition 4.1.A tree node τ of the branch and bound tree at depth d called **''unpromising**'' which accomplishes the subsequent condition:

$$s \leq s \text{ minutil } + \xi 1$$

where s minutil is the minimum utility score found and $\xi 1$ is a sub-optimality threshold which is accepted in the optimum result.

Definition 4.2. A ''future-worse-score'' node is a node which satisfies the following condition:
$s + \Delta min \leq s$ minutil $+ \xi 1$ where Δmin is the minimum estimated score that is going to be added to s.
The proposed method employs this as the lookahead utility pruning to avoid unnecessary computations. This prompts a great deal of utility calculations for itemsets that in the long run doesn't fulfill the minimum utility threshold.An itemset X is killed (along with its subtree) if it pleases the "future-worst-score" condition.

5. Pruning Stratagies

This section talks about an additional key issue for planning an well-organized pruned branch and bound calculation, which is to outline a powerful instrument for pruning. For this reason four new constraints on the itemsets utility called as total weighted utility, sub-tree utility ,local utility and depth impurity quality index are used.

5.1. TWU Based Pruning

Definition 5.1.1. Transaction-weighted itemset utility X,

$$TWU(X) = \sum_{X \subseteq T_j \in D} TU(T_j)$$

the TWU(A)=8+27+30=65. For any itemset X, if transaction-weighted utility is lesser than the specified minimum utility value at that time itemset X is considered as a low-utility along with all its supersets[9].They work in two stages.In the key stage, they recognize hopeful itemsets by figuring their transaction-weighted utility to trim the searching area. On the off chance that transaction-weighted utility of an itemset has more prominent than minimum utility value , it might stand and is hence designated as a hopeful itemset of high utility. In case an itemset has a transaction-weighted utility less than threshold as it could not be a HUI.

{a,b,c,d,e,f,g} ={65,61,96,58,88,30,38}

Table 4.Transaction List with items by TWU

TID	Transaction List	Purchase quantity	Utility values	Transaction utility values
T1	{d, a, c}	{1, 1, 1}	{2,5,1}	8
T2	{g, a, e, c}	{5,2,2,6}	{5,10,6,6}	27
T3	{f ,d, b, a ,e, c}	{5,6,2,1,1,1}	{5, 12,4,5,3,1}	30
T4	{d, b, e ,c}	{3,4,1,3}	{6,8,3,3}	20
T5	{g, b, e, c}	{2, 2, 1, 2}	{2,4,3,2}	11

Table 6 .TIDLIST

f

TID	U	RU
3	5	25

g

TID	U	RU
2	5	22
5	2	9

d

TID	U	RU
1	2	6
3	12	13
4	6	14

b

TID	U	RU
3	4	9
4	8	6
5	4	5

a

TID	U	RU
1	5	1
2	10	12
3	5	4

e

TID	U	RU
2	6	6
3	3	1
4	3	3
5	3	2

c

TID	U	RU
1	1	0
2	6	0
3	1	0
4	3	0
5	2	0

Algorithm 1:To create the High Utility list of K-itemset
Input: H.UList,the itemset H utility-list.
Hx.UList, the itemset Hx utility-list.
Hy.UList, the itemset Hy utility-list.
Output: Hxy.UList, the itemset Hxy utility-list.

1.Hxy.UList=Nil;
2.**for each** element Ex.Hx.UList **do**
3. **if** $\exists Ey \in Hy.UL$ ist and $Ex.tid == Ey.tid$ **then**
4 **if** H.UList ≠ empty **then**
//Utility list of K-temset {i1,i2,…ik-1, ik} ($k \geq 3$)
5.Do element search $E \in H.UL$ ist that E.tid ==Ex.tid;
6.Exy=<Ex.tid,Ex.i_util+Ey.i_util-E.i_util,Ey.r_util>;
7. **else** //(k=2)
8. Exy=<Ex.tid,Ex.i_util+Ey.i_util ,Ey.r_util>;
9. **end**
10. Append Exy to Hxy.UList;
11.**end**
12.**end**
13.**return** Hxy.UList;

Definition 5.1.2: An itemset X's remaining utility in transaction T_j($X \subseteq T_j$) is represented as

$$RU(X, T_j) = \sum_{X_i \in (Tj/X)} U(X_i, T_j)$$

For instance, in Table 5, collections of items T1 is T1/da=c.

5.2 Local Utility Based Pruning

Definition 5.2.1: Let α be an itemset and an item $Z \in E(\alpha)$. The Local Utility of Z with respect to α is

[1] $$lu(\alpha, Z) = \sum_{T \in g(\alpha \cup \{Z\})} [u(\alpha, T) + re(\alpha, T)].$$

For instance, when $\alpha = \{a\}$. The value of local utility for the items c, d and e are 65,30 and 57 respectively.If $lu(\alpha, Z) \prec \min util$, at that time all extension lead of α comprising Z is considered low-utility itemsets. In this case the item Z can be omitted while exploring all sub-trees of α. From all sub trees of an item set α is itemset and an item Z. some items may be removed from all sub-trees of an itemset, which decreases the quantity of itemsets to be treasured. In order to decrease the search area, furthermore, distinguish all sub-trees α that can be overlooked while investigating all sub-trees of α.

5.3 Sub-Tree Utility Based Pruning

According to the depth-first search $Z \in E(\alpha)$[1],the utility of sub-tree Z w.r.t. α

$$su(\alpha, Z) = \sum_{T \in g(\alpha \cup \{Z\})} \left[u(\alpha, T) + u(Z, T) + \sum_{i \in T \wedge i \in E(\alpha \cup \{Z\})} u(i, T) \right]$$

For instance, when $\alpha = \{b\}$ the value of subtree utility for the items c,d and e are 56, 4. and 12 respectively.If $su(\alpha, Z) \prec \min util$, then $\alpha \cup \{Z\}$ in the set-enumeration tree organization may be killed.

5.4 Depth-Impurity Quality Index Pruning

Depth impurity quality index pruning(DIQIP) reduces complexity of the grown tree by not letting the tree reach its full size. The cost complexity is measured by the Number of leaves and the depth in the tree. Fit a node to a leaf if Node depth is more striking than the tree depth threshold. This condition obtains high accuracy on unseen dataset can save a lot of calculation time involved in further growing the tree and then pruning it.

For a node Ω's quality is well-defined as a organization IQN (Ω) is relation to the tree T where Ω is placed. IQN_T (Ω) is given by

$$IQN_T(\Omega) = (1 - \phi(\Omega))\, f(depth_T(\Omega))$$

Where T is the decision tree which contains the node Ω and φ is an impurity measure stabilised between $[0,1]$.By presenting a damping function f , IQN takes the depth of a node within T(symbolized $depth_T$) , the performance quality may decrease when the search involves in a deeper node .

$$DI(T_s) = DI(\Omega_S) = \sum_{i=1}^{m} \frac{\alpha_i}{\alpha_s} IQN_T(\Omega'_i) \quad (2)$$

Equation (2) defines the quality index of a subtree T_S in T.DI is directly stemmed from IQN since the average of weights moves away Where $\Omega_1', \ldots, \Omega_m'$ are the leaves of T_S, α_i provides the number of examples in Ω_i' .Hence , Equation (2) may be written as

$$DI(T_S) = DI(\Omega_S) = \sum_{i=1}^{m} \frac{\alpha_I}{\alpha_S} (1 - \varphi(\Omega_i')) f(depth_T(\Omega_i')) \quad (3)$$

Referable to the definion of IQN, a leaf node's depth is calculated from the beginning of the complete decision tree.

It is prominent that the pruning, degree is rather certain to the improbability set with information. In applying, this implies the damping procedure has to be conformed in the data with the data as a way to collect, in all cases, an appropriate quantity of pruned trees. In this case a parameter is provided (denoted k) to standardize the process of damping , the higher the k, the more broad the pruning stage.Equation (4) provides the damping function restructured by this constraint.

The depth in the tree must be considered for the whole decision tree, which is often time consuming. Because while pruning a subtree, it is generally more reasonable to emphasis on the subtree compared to its root node. Therefore, the DI index can be defined in the following way,

$$DI(T_N) = \beta \sum_{M \in children_N} \frac{n_M}{n_N} DI(T_M)$$

$$f_k(d) = e^{-(k(d-1)/N)} with\ K \in R^+ \quad (4)$$

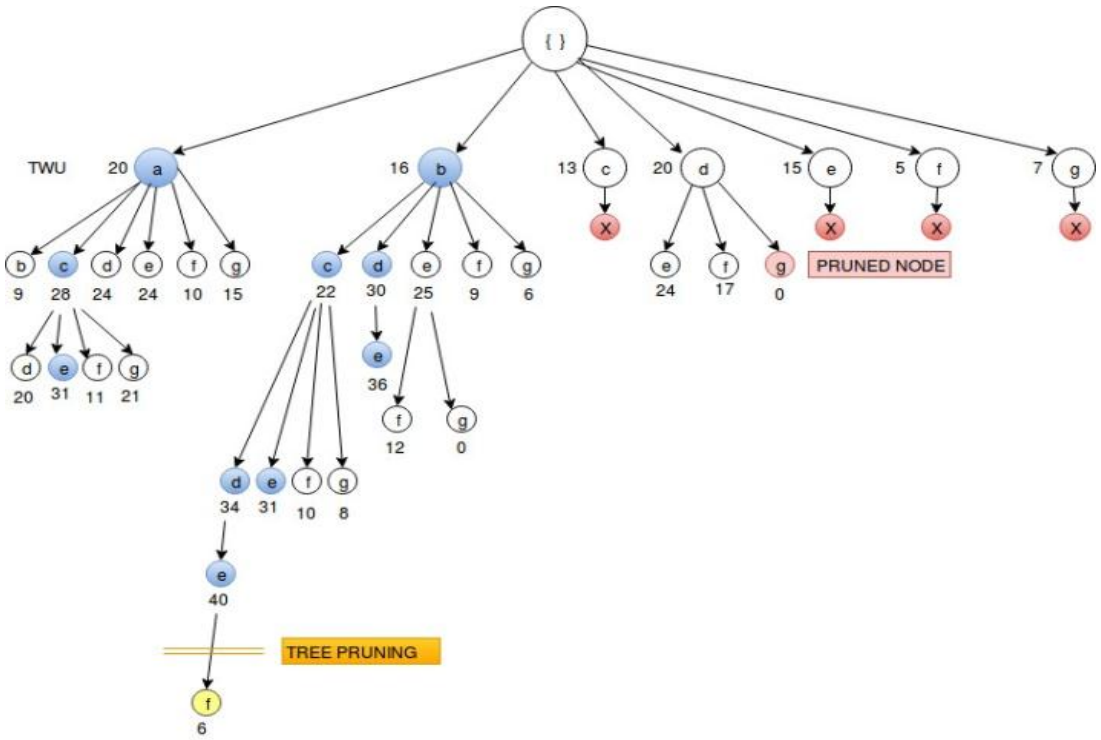


Fig.2. A model tree with design when depth impurity quality index pruning is applied

6. Complexity analysis

DIQIP algorithm can mine a database faster than finding the high utility itemset without pruning . Fig.3 shows that The runtime rate of DIQIP is low when the minimum utility value is large. Fig.4 shows that the runtime rate of DIQIP is high when the itemset is large. Because DIQIP algorithm takes almost as few levels as projected database , but without pruning method almost scans the same long length of database as minimum utility is small. In other words, when the minimum utility is large, there are fewer high utility itemset patterns.

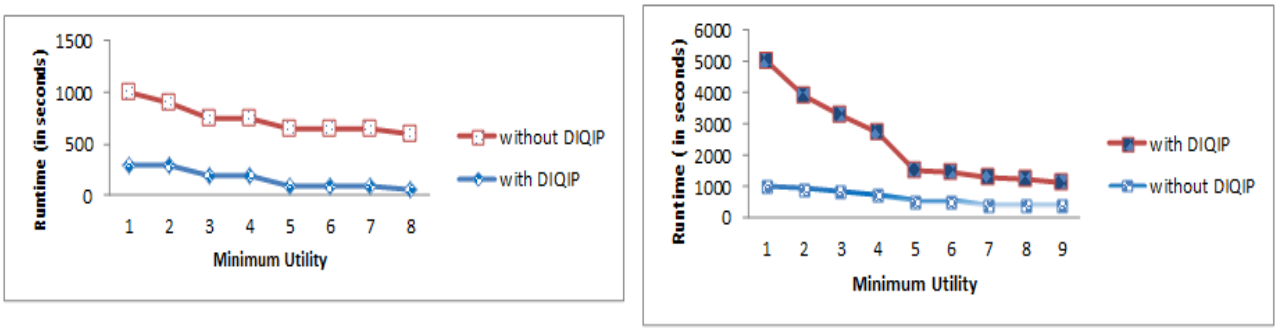
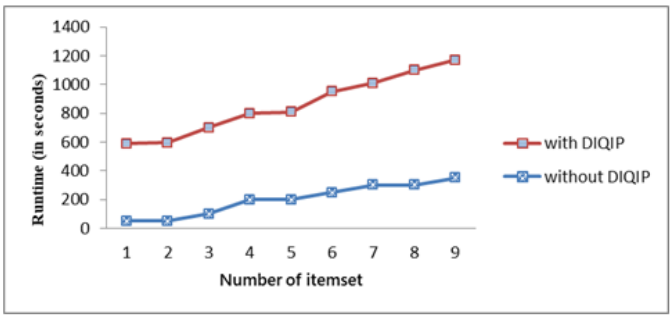


Fig.3. Run time Vs Minimum Utility



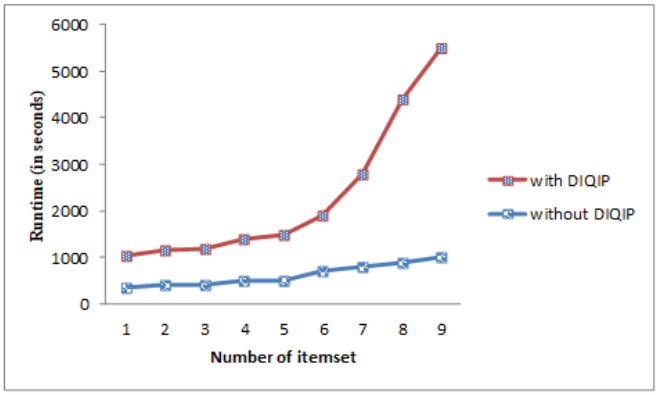


Fig.4. Run time Vs Number of itemset

7. Conclusion:

It is very important to optimize decision trees induction since data mining applications usually deal with very large databases. Depth impurity pruning stop the tree before full growing the tree can save time when compared to local utility pruning ,total weighted utility pruning and subtree utility pruning. Depth impurity pruning has better predictive accuracy in simple concepts. In learning complex concepts it is better to allow the tree to reach is full depth.

High sequential utility pattern mining can be viewed as an extension of high utility itemset mining, hence it can be explored more in the future.Mining high average-utility and strongly associated itemsets with combined criteria,therefore it can be explored more in the future.

References

[1] Lin, J. C. W., Ren, S., Fournier-Viger, P., & Hong, T. P. (2017). EHAUPM: Efficient High Average-Utility Pattern Mining With Tighter Upper Bounds. IEEE Access, 5, 12927-12940.

[2] Liu, Y., Liao, W.-k., &Choudhary, A. (2005). A two-phase algorithm for fast discoveryof high utility itemsets. In T. Ho, D. Cheung, & H. Liu (Eds.), Advances inKnowledge Discovery and Data Mining(pp. 689–695). volume 3518 of LectureNotes in Computer Science.

[3] Lin, J. C. W., Li, T., Fournier-Viger, P., Hong, T. P., Zhan, J., & Voznak, M. (2016). An efficient algorithm to mine high average-utility itemsets. Advanced Engineering Informatics, 30(2), 233-243.

[4] Li, Y.-C., Yeh, J.-S., & Chang, C.-C. (2008). Isolated items discarding strategy fordiscovering high utility itemsets. Data & Knowledge Engineering, 64, 198–217.

[5]Ahmed, C. F., Tanbeer, S. K., Jeong, B.-S., & Lee, Y.-K. (2011). Huc-prune: An efficientcandidate pruning technique to mine high utility patterns. Applied Intelligence,34, 181–198.

[6] Tseng, V. S., Shie, B.-E., Wu, C.-W., & Yu, P. S. (2012). Efficient algorithms for mining high utility itemsets from transactional databases. IEEE Transactions on Knowledge and Data Engineering, 25,1772–1786. http://dx.doi.org/10.1109/TKDE.2012.59.

[7]Liu, M., &Qu, J. (2012). Mining high utility itemsets without candidate generation.In Proceedings of the 21st ACM international conference on information andknowledge management CIKM '12 (pp. 55–64). New York, NY, USA: ACM.

[8] Liu, Y., Liao, W.-k., &Choudhary, A. (2005). A two-phase algorithm for fast discoveryof high utility itemsets. In T. Ho, D. Cheung, & H. Liu (Eds.), Advances inKnowledge Discovery and Data Mining(pp. 689–695). volume 3518 of LectureNotes in Computer Science

[9] Lan, G.-C., Hong, T.-P., Tseng, V. S., et al. (2012). An efficient gradual pruningtechnique for utility mining. International Journal of Innovative ComputingInformation and Control, 8, 5165–5178

[10]Lan, G.-C., Hong, T.-P., & Tseng, V. S. (2014). An efficient projection-based indexingapproach for mining high utility itemsets. Knowledge and Information Systems,38, 85–107

[11] Erwin, A., Gopalan, R. P., &Achuthan, N. (2007). A bottom-up projection basedalgorithm for mining high utility itemsets. Proceedings of the 2nd internationalworkshop on Integrating artificial intelligence and data mining (vol.84, pp. 3–11).Australian Computer Society, Inc..

[12] Wu, R. & He, Z.(2018).Top-k high average-utility itemsets mining with effective pruning strategies, Appl Intell (2018). https://doi.org/10.1007/s10489-018-1155-9.

[13] Liu, J., Wang, K., & Fung, B. (2012). Direct discovery of high utility itemsets withoutcandidate generation. In Proceedings of the 2012 IEEE 12th internationalconference on data mining (pp. 984–989). IEEE Computer Society

[14] KrishnamoorthyS(2015) Pruning strategies for mining high utility itemsets.Expet Systems with Applications,42(5):2371-2381

[15] Fournier–Viger P, Wu CW, Zida S,T seng VS (2014)FHM:Faster high-utility itemset mining using estimated utility co-occurrence pruning.Proceedings of the 21st International Symposium on Methodologies for intelligent systems, Roskilde, Denmark, June 2014.Lecture Nots in Artificial Intelligence 9384,Springer, Berling, oo.83-92

[16] Shao J, Meng X, Cao L (2016) Mining actionable combined high utility incremental and associated patterns. In: Ieee/csaa International conference on aircraft utility systems, pp 1164–1169